# Lab 1: .NET 3.5 Console Application Client for caBIO caGrid Service

*caBIG 2009 Annual Meeting Hack-a-thon*

University of Virginia eScience Group
Marty Humphrey, Director

*Overview:* Create the simplest of .NET applications as a client for the caBIO caGrid Service. A graphical wizard built into Visual Studio is used to build client-side proxy objects for the caBIO caGrid Service. Because of minor interoperability issues, we must edit one of the auto-downloaded files from the caBIO Grid Service and then regenerate the proxy objects by hand. We then insert a CQL query in the main body of the client. In Part 1 of this lab, we just print to the screen the XML representation of the caBIO objects returned from the service. In Part 2, we deserialize the objects into C# representations.

*Expected duration*: 30 minutes

## Part 1: Printing the XML returned from caBIO to the screen

1) Run Visual Studio 2008 (Start → All Programs → Microsoft Visual C# 2008 Express Edition)
   a) Create a new C# project: console app (File → New Project →Console Application). Name it caBIOClient and hit "OK" in this "New Project" window.
   b) Right click on caBIOClient in the "Solution Explorer" (upper right) and "Add Service Reference". Type into the "Address" field: http://cabiogrid40.nci.nih.gov/wsrf/services/cagrid/CaBIO40GridSvc?wsdl and hit "Go". Change the "Namespace" to be "caBIOSvc" and then hit "Okay".
   c) Quit Visual Studio (make sure to "Save" when prompted).
2) Almost all of the WSDL/XSD retrieved by "Add Service Reference" in the previous step is ready for use by the Microsoft 3.5 proxy-generation tooling, but one file is not – in this step, we'll edit this WSDL file by hand.
   a) Start → My Documents and then navigate to the "Service References" folder of this project (e.g., My Documents\Visual Studio 2008\Projects\caBIOClient\caBIOClient\) and then open the "caBIOSvc" folder.
   b) Double-click on CaBIO40GridSvc2.wsdl, which will open the file in Visual Studio 2008. Make the following changes (Note that the bottom of Visual Studio 2008 will indicate what line you're currently editing). When done, hit Ctrl-S to save the file. Exit Visual Studio 2008.

| ***Line number*** | ***Old contents*** | ***New contents*** |
|---|---|---|
| 9 | <wsdl:input> | <wsdl:input name="GetMultipleResourcePropertiesRequest"> |
| 12 | <wsdl:output> | <wsdl:output name="GetMultipleResourcePropertiesResponse"> |
| 24 | <wsdl:input> | <wsdl:input name="GetResourcePropertyRequest"> |
| 27 | <wsdl:output> | <wsdl:output name="GetResourcePropertyResponse"> |
| 39 | <wsdl:input> | <wsdl:input name="QueryResourcePropertiesRequest"> |
| 42 | <wsdl:output> | <wsdl:output name="QueryResourcePropertiesResponse"> |

3) Start up a Windows command prompt (Start → All Programs → Accessories → Command Prompt)
   a) Cd to the folder in Step 2a, above
   b) Run the following to generate the proxy code: "c:\Program Files\Microsoft SDKs\Windows\v6.0A\bin\SvcUtil.exe" *.wsdl *.xsd

c) Four "validation errors" will appear on the screen, which you can ignore. This step will generate 2 files: BaseFaults.cs and output.config, which are both automatically saved in this folder. We'll use these in a moment. ***If the last line printed out on the screen does NOT mention "output.config", then you probably have a typo (note that case matters) in the 6 modifications described in Step 2b.***

d) Execute "cd ..\.." to get out of this folder (but do not kill this command prompt)

4) Kill off the Windows Explorer (from Step 3) that is currently in "caBIOSvc"

5) Re-Run Visual Studio 2008 (Start → All Programs → Microsoft Visual C# 2008 Express Edition)

a) Open up the caBIOClient Solution (it should be the first listed under the "Recent Projects")

b) Add the *two* files we just created:

i) right-click on caBIOClient project in the Project Explorer -- it's the line immediately below the "Solution 'caBIOClient' (1 solution)", select "Add → Existing Item" and navigate to "caBIOsvc" folder (in step 2a) and then select BaseFaults.cs

ii) Do this "Add → Existing Item" again, change the dialogue's "Objects of type" to be "All Files", and add "output" (note: this is actually the "output.config" file, but "config" may not be shown in the window)

c) Right-click on "app.config" and select "delete"

d) Right-click on the "caBIOSvc" Service Reference and select "delete"

e) Right-click on "BaseFaults.cs" and select "rename" to "caBIOSvc.cs"

f) Right-click on "output.config" and select "rename" to "app.config"

g) Now we're ready to add the real client functionality, by double-left-clicking on "Program.cs" and adding into the body of main (cut-and-paste this code). The explanation of this code is:

i) In the first line, we creates an object that acts as the local proxy to the remote service (this service is defined in "caBIOSvc.cs" and "app.config")

ii) The next lines create the CQL query to ask the service for the number of chromosomes it has and then invokes the "query" operation on the service

iii) The final three lines interpret the results from the service as a "CQLCountResult" and prints the result to the screen

```
CaBIO40GridSvcPortTypeClient proxy = new CaBIO40GridSvcPortTypeClient();
QueryRequestCqlQuery arg = new QueryRequestCqlQuery();
arg.CQLQuery = new CQLQuery();
arg.CQLQuery.Target = new Object();
arg.CQLQuery.Target.name = "gov.nih.nci.cabio.domain.Chromosome";
arg.CQLQuery.QueryModifier = new QueryModifier();
arg.CQLQuery.QueryModifier.countOnly = true;
CQLQueryResults result = proxy.query(arg);

CQLCountResult cr = (CQLCountResult)result.Items[0];
Console.WriteLine("query returned {0} results", cr.count);

Console.ReadLine();
```

h) Ctrl-S and hit F5 to execute. If successful, you will see the following on the screen:

query returned 84 results

i) To show the actual XML representation instead of the count of Chromosomes, *comment-out the two lines of code pertaining to the "QueryModifier",* replace the second and third lines from the end with the following, then hit Ctrl-S and hit F5 to execute:

```
 for (int i = 0; i < result.Items.Length; ++i)
{
    CQLObjectResult objRes = (CQLObjectResult)result.Items[i];
    Console.WriteLine(objRes.Any.OuterXml);
}
```

## Part 2: Generating C# objects by Deserializing the XML returned from the caBIO caGrid Service

6) Replace (cut-and-paste) the entire body of main with this code (it is very similar to the existing body but contains enough differences such that it's easiest just to replace the body entirely). The explanation of this code is:
   a) The first part of this code creates the proxy object, constructs the CQL query, and interacts with the service.
   b) The middle part creates a list ("chromos") of the Chromosome objects by looping through the list of XML representations of the chromosomes returned by the service. For each, we use a C# "deserializer" to create the Chromosome object and then add the object to the "chromos" list. *Note that we got the definition of "Chromosome" automatically when we added the "Service Reference" in Step 1b*
   c) The last part prints the "id" and "bigid" of each chromosome to the screen

```
CaBIO40GridSvcPortTypeClient proxy = new CaBIO40GridSvcPortTypeClient();
QueryRequestCqlQuery arg = new QueryRequestCqlQuery();
arg.CQLQuery = new CQLQuery();
arg.CQLQuery.Target = new Object();
arg.CQLQuery.Target.name = "gov.nih.nci.cabio.domain.Chromosome";
CQLQueryResults result = proxy.query(arg);

List<Chromosome> chromos = new List<Chromosome>();
for (int i = 0; i < result.Items.Length; ++i)
{
    CQLObjectResult objRes = (CQLObjectResult)result.Items[i];
    Console.WriteLine(objRes.Any.OuterXml);

    XmlNodeReader xnr = new XmlNodeReader(objRes.Any);

    XmlSerializer xs = new XmlSerializer(typeof(Chromosome),
           "gme://caCORE.caBIO/4.0/gov.nih.nci.cabio.domain");

    chromos.Add((Chromosome)xs.Deserialize(xnr));
    xnr.Close();
}

Console.WriteLine("\nID and Big ID of each chromosome:\n");

foreach (Chromosome chr in chromos)
{
    Console.WriteLine(chr.id + ": " + chr.bigid);
}

Console.ReadLine();
```

7) Add the following lines toward the top of the file:

```
using System.Xml;
using System.Xml.Serialization;
```

8) Ctrl-S and hit F5 to execute. If successful, you should see the following on the screen:

<ns2:Chromosome bigid="hdl://2500.1.PMEUQUCCL5/ULRNAG52YB" number="*" id="91"
xmlns:ns2="gme://caCORE.caBIO/4.0/gov.nih.nci.cabio.domain" />
<ns3:Chromosome bigid="hdl://2500.1.PMEUQUCCL5/HHLOPWRKP3" number="14_random" id="84"
xmlns:ns3="gme://caCORE.caBIO/4.0/gov.nih.nci.cabio.domain" />
<ns4:Chromosome bigid="hdl://2500.1.PMEUQUCCL5/65PGWADFV3" number="M" id="85"
xmlns:ns4="gme://caCORE.caBIO/4.0/gov.nih.nci.cabio.domain" />

*… some chromosomes deleted in Lab1 printout for brevity …*

<ns84:Chromosome bigid="hdl://2500.1.PMEUQUCCL5/333I2BJDOE" number="U" id="90"
xmlns:ns84="gme://caCORE.caBIO/4.0/gov.nih.nci.cabio.domain" />
<ns85:Chromosome bigid="hdl://2500.1.PMEUQUCCL5/EN6UOXEIYT" number="*" id="92"
xmlns:ns85="gme://caCORE.caBIO/4.0/gov.nih.nci.cabio.domain" />

ID and Big ID of each chromosome:

91: hdl://2500.1.PMEUQUCCL5/ULRNAG52YB
84: hdl://2500.1.PMEUQUCCL5/HHLOPWRKP3
85: hdl://2500.1.PMEUQUCCL5/65PGWADFV3
86: hdl://2500.1.PMEUQUCCL5/AY2IZQZOBI

*… some chromosomes deleted in Lab1 printout for brevity …*

82: hdl://2500.1.PMEUQUCCL5/AH357BTPLM
83: hdl://2500.1.PMEUQUCCL5/W6GFBF7NC2
87: hdl://2500.1.PMEUQUCCL5/AGXX7SOTTJ
90: hdl://2500.1.PMEUQUCCL5/333I2BJDOE
92: hdl://2500.1.PMEUQUCCL5/EN6UOXEIYT